Strukturieren Sie ihre Anforderungen (noch) mit Kapitelstrukturen? Ein Plädoyer für architekturgetriebenes Requirements Engineering

Dr.-Ing. Oliver Alt
Continental
Division Chassis & Safety, Electronic Brake Systems
Guerickestrasse 7, D-60488 Frankfurt am Main
oliver.alt@continental-corporation.com

1 Einleitung

War das Thema und die Tätigkeit des Requirements Engineering (RE) bis vor wenigen Jahren vor allem noch im Bereich der Softwareindustrie und der Forschung beheimatet, hat es sich - nicht zuletzt begünstigt durch Entwicklungsnormen, wie z.B. AutomotiveSPICE [MHDZ07]- inzwischen auch in der übrigen Industrie breit etabliert. Neben der Anforderungsanalyse und dem Anforderungsmanagement ist auch System- und Softwarearchitektur essentieller Teil eines modernen Entwicklungsumfeldes. Leider werden das Requirements Engineering und die Architektur sehr häufig unabhängig voneinander gesehen und auch so in der Praxis umgesetzt.

Dieser Beitrag zeigt, wie man in einem modernen Entwicklungsumfeld Architektur und Requirements Engineering so koppeln kann, dass diese sich direkt ergänzen. Dabei entwickeln und strukturieren sich die Anforderungen anhand der Architektur und umgekehrt.

2 Dokumenten- vs. Modell-basierte Entwicklung

In der „klassischen“ dokumentenbasierten Systementwicklung stehen Dokumente als Arbeitsprodukte im Vordergrund. Diese werden zumeist von verschiedenen Bearbeitern erstellt und verteilt. Ein Problem entsteht dann, wenn man Inhalte aus verschiedenen Dokumenten miteinander verknüpfen muss, zum Beispiel zur Herstellung einer Nachverfolgbarkeit (Traceability) zwischen Anforderungen und Architektur.

Eine Lösung für dieses Problem bietet der Einsatz von Modell-basierter Entwicklung (vgl. [Alt10]). Dabei tritt ein datenbankgestütztes Modell an die Stelle der Dokumente. In dieses Modell arbeiten die Architekten und Requirement-Ingenieure gleichermaßen ihre Entwicklungsdaten zum Zweck einer Systemspezifikation ein.

Eine grafische Sprache, um solch ein Systemmodell zu erstellen, ist die Systems Modeling Language (SysML) [Wei06]. Mit SysML ist es möglich 1. Strukturen eines Systems, also die Architektur, 2. das Systemverhalten und 3. die Anforderungen zu modellieren. Da sich all diese Inhalte in einer gemeinsamen Datenbank befinden, können sie beliebig miteinander verknüpft werden und somit auch die Anforderungen an Tracability leicht erfüllen.

Dokumente lassen sich aus einem solchen Modell automatisch generieren, da die Daten dort vorhanden sind und mit Hilfe von Werkzeugen verarbeitet werden können.

3 Architekturgetriebene Anforderungsentwicklung

Wie sollte man nun vorgehen um Anforderungen anhand der Architektur zu entwickeln? Mit dem Bekenntnis zu Modell-basierter Entwicklung ist es nicht getan. Man braucht eine entsprechende Methodik in der Anwendungs- und Modellierungsregeln definiert werden.

Eine entsprechende Methodik wurde in den vergangenen drei Jahren bei Continental entwickelt. Diese wird Anhand eines einfachen Beispiels nun erläutert. Dabei soll ein Thermometer entwickelt werden, das seine Umgebungstemperatur aufnimmt und diese dem Benutzer als Text anzeigt.

Die entwickelte Methodik stützt sich zunächst darauf, dass die Architektur eines technischen Systems mit Hilfe von so genannten *Wirkketten* dargestellt wird. In einer Wirkkettendarstellung werden die Komponenten des Systems, bestehend aus Hard- und Software in einer funktionalen Kette von Komponenten modelliert.

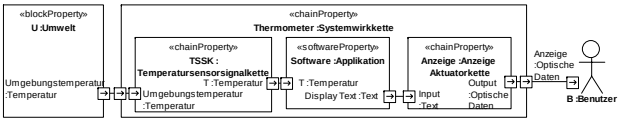

Abbildung 1: Die oberste Architekturebene als Wirkkette

Abbildung 1 zeigt die oberste Architekturebene des Thermometers als Wirkkette. Am Beginn der Kette geht die Temperatur als Signal der Umwelt in das System ein, wird dort verarbeitet und am Ende der Kette dem Benutzer des Thermometers als optische Information übergeben.

Die Darstellung der Architektur als Wirkkette unterscheidet sich von der klassischen *physikalischen Architekturdarstellung*, bei der alle physikalisch sicht- und greifbaren Komponenten dargestellt werden. Diese physikalische Architektur entsteht dadurch, dass man das Gerät oder System, das man beschreibt im Prinzip zerlegt und das darstellt was man sieht.

Auch diese Architektursicht braucht man, jedoch ergibt sich das Gesamtbild des Systems erst in Kombination mit der Wirkkettendarstellung. Die physikalische Darstellung hat nämlich einen entscheidenden Nachteil: Softwarekomponenten lassen sich nicht darstellen, da sich die Komponentenstruktur der Software spätestens mit dem Übersetzungsvorgang auflöst. Physikalisch gesehen ist Software nur noch als Ladungsträgerverteilung im Speicher des

Systems vorhanden. Daher wird die Wirkkettenarchitektur benötigt um die logischen Zusammenhänge zwischen Hard- und Software zu modellieren.

Auch innerhalb der Komponente *Thermometer:Systemwirkkette* wird wiederum von der Kettendarstellung Gebrauch gemacht. Es gibt eine Temperatursensorkette und eine Anzeige Aktuatorkette. Diese sind funktionale Einheiten, die die Aufgabe haben als Bindeglied zwischen der Applikationssoftware und der Systemumgebung zu fungieren.

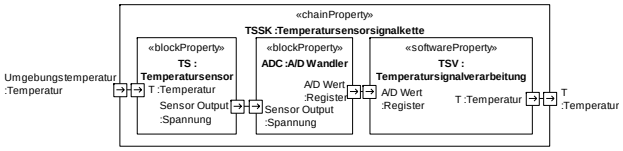

Abbildung 2: Innere Architektur der Temperatursensorkette

In Abbildung 2 ist die innere Struktur der Temperatursensorkette dargestellt. Sie besteht aus einem Sensor als Wandlerelement, einer Elektronikkomponente in Form eines A/D-Wandlerkanals und einer Low-Level-Softwarekomponente die das A/D-Wandlerregister ausliest und die Temperatur als Softwaresignal am Ausgang der Kette bereit stellt. Die Anzeige Aktuatorkette auf der Ausgangsseite funktioniert genau umgekehrt und besteht daher aus Software, Elektronik und Elektro-Optischem-Wandler (Display).

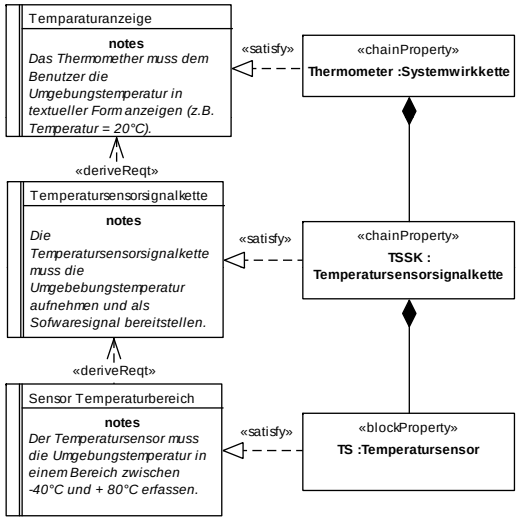

Abbildung 3: Anforderungsherleitung anhand der Architektur

Mit Hilfe dieser Architekturmuster lassen sich nun auf einfache Art und Wiese Anforderungen herleiten und verknüpfen. In Abbildung 3 ist das Ergebnis eines solchen Herleitungsprozesses mit Hilfe von SysML-Notation dargestellt. Dieses Diagramm wurde dabei schrittweise entwickelt. Am Anfang stand nur die oberste Architekturkomponente (*Thermometer:Systemwirkkette*) und deren Anforderung. Aufgrund dieser Anforderung wurde die Architekturentscheidung getroffen die Aufnahme der Umgebungstemperatur mit Hilfe einer Sensorkette zu lösen. Als nächstes ergab sich die Anfordeung an diese Komponente. Dies wurde fortgesetzt bis auch der Sensor als Architekturschritt und dessen Anforderung entwickelt war.

Die Architekturkomponenten sind über eine Besteht-Aus-Beziehung (Verbindung mit Raute) miteinander verknüpft. Ebenso wurden die Anforderungen mit satisfy-Beziehungen der Architektur zugeordnet. Letztendlich ergibt sich noch die deriveReqt-Beziehung als Modellierung der Anforderungsableitung praktisch fast automatisch aufgrund der Architekturzerlegung. Damit entsteht ein vollständig geschlossener Kreis von Verknüpfungen zwischen Architekturelementen und Anforderungen.

4 Fazit

Anforderungen und Architektur gehen Hand in Hand. Aus Anforderungen ergibt sich eine Architektur als Lösung und damit ergeben sich neue Anforderungen an die nun definierten Komponenten und deren Verfeinerung.

In jeder Anforderung steckt im Text implizit ein Architekturverweis, wenn man schreibt *Das **System** muss...* oder *Der **Sensor** muss....* Damit kann man keine Anforderung schreiben, ohne ein Bild der Architektur bereits (im Kopf) erstellt zu haben, und sei es zu Beginn nur ein Kasten, der mit ***System*** bezeichnet wird.

Mit Modell-basierter Entwicklung lassen sich die Anforderungen architekturgetrieben entwickeln und strukturieren. Dies führt zu einer geschlossenen Nachverfolgbarkeit und einer Top-Down-Arbeitsweise. Thematische Gliederungen können auch weiterhin ergänzend verwendet werden, so dass sich „alte“ Dokumenten- und „neue“ Modellwelt sinnvoll kombinieren.

Der Ansatz löst viele Probleme aus der Praxis des Requirements Engineering, die dadurch entstehen, dass Anforderungen im falschen Kontext entwickelt und geschrieben werden und dadurch unvollständig oder falsch zugeordnet werden. In einem Umfeld wie der Automobilzulieferindustrie, wo Systeme und Anforderungen Top-Down entwickelt werden müssen, um als Eingangsinformation für die folgende Subsystem-, bzw. Komponentenentwicklung zu dienen, ist es wichtig Anforderungen nachvollziehbar und strukturiert zu entwickeln. Hier hilft der Ansatz enorm und wird in der Praxis bereits in mehreren Projekten erfolgreich eingesetzt.

Literatur

[Alt10] O. Alt. Vollständige Traceability durch modellbasierte Entwicklung mit SysML erreichen. In *Proceedings of iqnite 2010, Düsseldorf*, 2010. www.iqnite.com.

[MHDZ07] M. Müller, K. Hörmann, L. Dittmann und J. Zimmer. *Automotive SPICE in der Praxis - Interpretationshilfe für Anwender und Assessoren*. dpunkt.verlag, 2007.

[Wei06] Tim Weilkiens. *Systems Engineering mit SysML/UML : Modellierung, Analyse, Design*, Jgg. 1. Aufl. dpunkt-Verlag, Heidelberg, 2006.